Fundamentos de Java

Prof. Marcelo Cohen

1. Histórico

- 1990
  - linguagem Oak;
  - desenvolvimento de software embutido para eletrodomésticos
  - S.O. para o controle de uma rede de eletrodomésticos
  - o surgimento da Web redirecionou Oak dando origem a Java
- 1995
  - A linguagem Java foi disponibilizada pela 1ª vez: JDK 1.0
  - Adoção de Java na Web: segurança → applets rodam em um ambiente controlado (browsers)

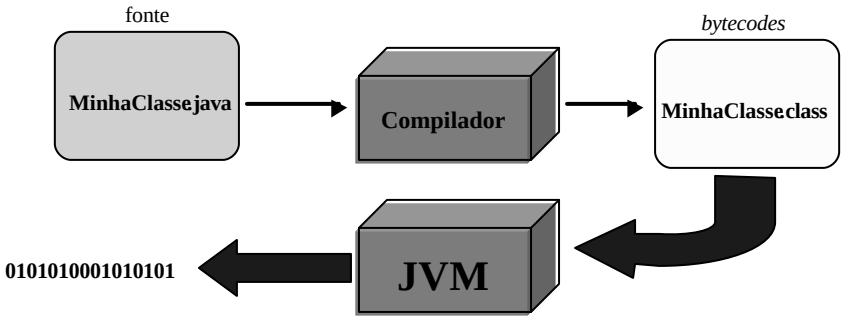
- 1997
  - JDK 1.1
  - Nova forma de tratar os eventos (*listeners*)
  - Componentes (*beans*)
  - JDBC (acesso a banco de dados)
  - Swing (modelo de interface independente de plataforma)
  - Java 3D
  - Ambientes de programação: Visual J++, Jbuilder, Visual Café, Kawa, Java Workshop
  - Crescimento da linguagem (parte do público Web migrou para soluções mais simples JavaScript e VBScritp)
  - Surgimento de compiladores (Symantec e SuperCede)
- 1999
  - JDK 1.2 ou Java 2
  - Correção de Bugs, otimização sedimentação dos conceitos

2. Características

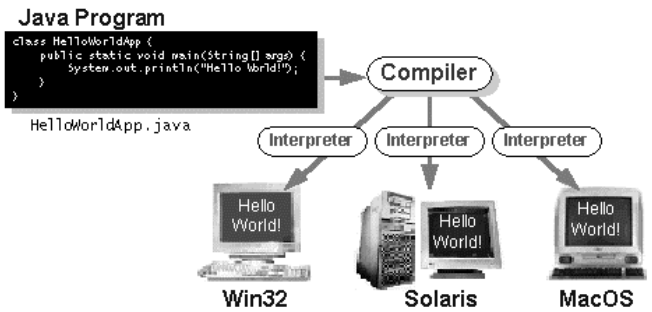
- Java é tanto uma linguagem de programação de alto nível quanto uma plataforma.

- Como linguagem, Java é:
  - orientada a objetos
  - independente de arquitetura (portável)
  - robusta
  - segura
  - interpretada
  - distribuída

- Java é tanto compilada como interpretada:
  - compilador transforma o programa fonte em **bytecodes**
  - **Bytecodes** são instruções compreendidas pela *Máquina Virtual Java*
  - A Máquina Virtual Java *(JVM)* é um interpretador, que transforma as instruções em linguagem de máquina
  - *"Write once, run anywhere"* - slogan criado pela Sun, para demonstrar a portabilidade da linguagem (graças aos *bytecodes*)



- Java é tanto compilada como interpretada:
  - compilador transforma o programa fonte em **bytecodes**
  - **Bytecodes** são instruções compreendidas pela *Máquina Virtual Java*
  - A Máquina Virtual Java *(JVM)* é um interpretador, que transforma as instruções em linguagem de máquina
  - *"Write once, run anywhere"* - slogan criado pela Sun, para demonstrar a portabilidade da linguagem (graças aos *bytecodes*)



- Como plataforma, Java compreende uma JVM e uma API (*application programming interface)*.
  - Programas podem ser executados como aplicações tradicionais ou em páginas web.
  - Applications - são executados pelo sistema operacional e podem ser
    - console applications: quando não apresentam saída gráfica, somente textual
    - windowed applications: criam e gerenciam múltiplas janelas, usam mecanismos de GUI (*graphical user interface*) para a programação.
  - Applets - são programas executados pelo navegador Web, através de uma JVM própria (interna)
    - a característica principal dos applets é a utilização da própria área da página como interface
    - applets são executados em um ambiente restrito, oferecendo segurança

- Outras considerações:
  - Para se ter flexibilidade e segurança, abre-se mão da **velocidade de execução**.
  - Um programa Java típico roda cerca de 10 vezes mais lentamente do que um programa equivalente compilado em código nativo.
  - Para resolver esse problema, foi criado um sistema de compilação em tempo de execução, denominado **JIT - just-in-time compilation.**
  - Um compilador JIT compila uma classe em código nativo no momento em que esta é lida para a memória.
  - A carga dos programas torna-se mais lenta, mas o ganho de velocidade compensa.

- Futuro ???
  - Idéia principal: portabilidade
  - No futuro, vai ser possível integrar diversos dispositivos diferentes, por exemplo, ligar um computador a uma torradeira, ambos rodando programas em Java.

  - Está em desenvolvimento um *chip* (JavaChip), que poderá executar *bytecodes* diretamente. Um *chip* como esse poderá equipar diversos aparelhos domésticos.
  - Já existem protótipos de sistemas de navegação para veículos, sistemas embutidos, *smart cards*, vídeo-fones, etc.
  - Na área de negócios, a possibilidade de integrar diversos sistemas diferentes à Internet é uma grande vantagem.

3. Estrutura de um programa em Java

- Um programa é composto por uma ou mais classes.
- Tipicamente, cada classe é escrita em um arquivo fonte separado, cujo nome deve ser o mesmo da classe, com o sufixo .java.
  - Ex: a classe “Pilha” deverá estar armazenada no arquivo **Pilha.java**
- Em geral, todas as classes que compõem um programa deverão estar no mesmo diretório.

- Biblioteca de classes Java
  - Da mesma forma que a biblioteca de funções da linguagem C, a biblioteca de classes armazena uma coleção de classes de uso geral, para as tarefas mais comuns em programação.
  - Classes são agrupadas em conjuntos, denominados pacotes (*packages*).
  - Exemplos de pacotes:
    - **java.lang** - inclui classes básicas, manipulação de arrays e strings. Este pacote é o único que é carregado automaticamente por qualquer programa.
    - **java.io** - classes para entrada e saída de dados
    - **java.util** - classes diversas para manipulação de dados
    - **java.applet** - utilizadas para a implementação de applets
    - **java.awt** - utilizadas para aplicações baseadas em janelas

4. Ambiente de Desenvolvimento e Execução

- É necessário instalar o kit para desenvolvimento de software Java, ou JDK (*Java Development Kit)*.
- O JDK oferece diversas ferramentas, mas as fundamentais são:
  - **javac** - compilador: gera bytecodes a partir de código-fonte
  - **java** - interpretador: interpreta (ou compila, se suportar JIT) os bytecodes para linguagem de máquina

Demonstração !